Improved Color Modeling in Different Color Spaces

Anonymous EMNLP-IJCNLP submission

Abstract

We present a model that grounds color comparative adjectives in 2 different color spaces. We find that modifiers represented as vectors from reference colors to target colors show different behaviors in RGB and HSV color space. Based on this finding we design models that primarily improve modeling of color related modifiers, such as "pinkish". In experiments, we pre-train basic models in different color spaces and train hard and soft ensemble models. Experimental results show significant and consistent improvements compared to the state-of-the-art baseline model.

1 Introduction

Comparative color descriptions are employed to describe colors which are not covered by basic colors terms (Monroe et al., 2017), for instance, "greenish blue" cannot be expressed by only "blue" or "green". Comparative descriptions are essential for image captioning (Karpathy and Fei-Fei, 2015), object recognition (van de Sande et al., 2010) and other grounded language understanding problems.

In this paper, we present models that are able to predict a target color given a reference color and a modifier. For example, for the triple ("red", "pinkish", "pinkish red") in RGB space, r, m and t are three strings which refer to "red", "pinkish" and "pinkish red" respectively. In addition, vectors $\vec{r}$ and $\vec{t}$ are the corresponding RGB codes for r and t. According to Munroe (2010), in this example $\vec{r} = \begin{bmatrix}229 & 0 & 0\end{bmatrix}^\top$ and $\vec{t} = \begin{bmatrix}241 & 12 & 69\end{bmatrix}^\top$.

The recent state-of-the-art model (Winn and Muresan, 2018) learns a vector $\vec{m}$ as a function of $\vec{r}$ and m such that $\vec{t} = \vec{r} + \vec{m}$. We note that this model simply represents a modifier as a set of vectors with almost the same direction in different situations, and as a result, it fails to model modifiers with complex transformations, for example, color related modifiers, like "greenish".

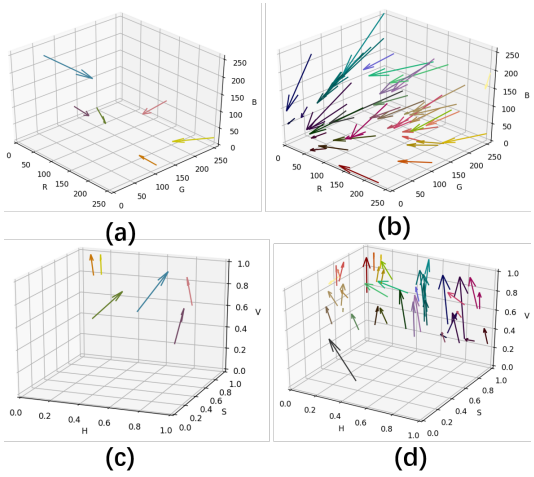


Figure 1: Each arrow refers to a $\vec{m}$. (a) and (c) show $\vec{m}_i$ in RGB space and HSV space where $i \in \{\, i \mid m_i =$ "dirty"$\}$. Similarly, (b) and (d) show $\vec{m}_j$ in RGB space and HSV space where $j \in \{\, j \mid m_j =$ 'dark"$\}$.

As shown in Figure 1, vectors show local convergence in RGB space and local parallels in HSV space. That is, their behavior is correlated to r in each instance. Since $\vec{m}$ shows different patterns in RGB and HSV, we present the two types of models: RGB model and the HSV model. In addition, combining the RGB model and HSV model, we trained a hard ensemble model to select which space should be used for prediction given a certain $(\vec{r}, m)$ pair.

Experiments show that our hard ensemble model achieves state-of-the-art performance in terms of both Cosine similarity and the Delta-E evaluation metric.

2 Methods

Here we describe the methods employed for color modeling. Formally, the task is to predict a target

color t in RGB space from a reference color r and a modifier m. These three objects are represented as vectors which we call $\vec{t}, \vec{r}$ and $\vec{m}$, respectively.

2.1 Baseline

Winn and Muresan (2018) present a model (WM18) which represents a vector $\vec{m} \in \mathbb{R}^3$ as a function of $(m, \vec{r})$ pointing from a reference color vector $\vec{r}$ to the target color vector $\vec{t}$, such that $\vec{t} = \vec{r} + \vec{m}$. In the simplest case the modifier m does not depend on the reference color r and can thus be represented by the same vector for all colors. This assumption, however, does not hold in all situations. For example, when predicting an instance with "greenish" as the modifier, WM18 get a $\vec{m}$ with cosine similarity -0.76 which is in the almost opposite direction.

In order to resolve this problem, we model modifiers in both RGB and HSV space and train a model for color space selection.

Diagonal Covariance For our Diagonal Covariance model(DC), instead of being modeled as a vector from $\vec{r}$ to $\vec{t}$, $\vec{m}$ which is a function of m is represented as a point in RGB space. Given $\vec{r}$ and m, to predict the $\vec{t}$, $\vec{m}$ is predicted first. Based on the prediction, we model the target as

$$\vec{t} = \vec{r} + \alpha_m \times (\vec{m} - \vec{r}), \tag{1}$$

where $\alpha_m \in [0, 1]$ is a scalar which only depends on m and measures the distance from $\vec{r}$ to $\vec{m}$. Given the error term $\varepsilon \sim \mathcal{N}(\vec{0}, \sigma I_{3\times3})$, for any α_m, the probability density function of the Gaussian distribution of the $\vec{t}$ is as follows:

$$f(\vec{t_i}) = \mathcal{N}(\vec{t_i}; \vec{\mu_i}, \sigma I_{3\times3}), \tag{2}$$

where $\vec{\mu_i} = \vec{r} + \alpha_m \times (\vec{m} - \vec{r})$ (see Equation (1)).

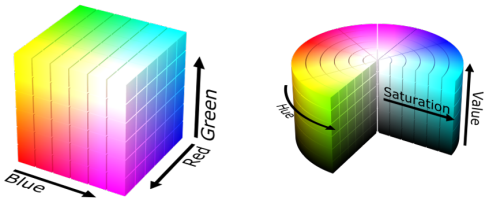
Thus DC is trained to minimize the following loss:

$$\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n}(\vec{t_i} - \hat{t_i})^\top(\vec{t_i} - \hat{t_i}) \tag{3}$$

where $\vec{t_i}$ is the target vector in each instance and $\hat{t_i}$ is the prediction.

General We note that vectors $\vec{m}_i$ are not strictly convergent. To overcome this problem, we propose a general model in RGB space:

$$\vec{t} = M\vec{r} + \vec{\beta} \tag{4}$$



(a) RGB space (a) HSV space

Figure 2: a) the RGB color space. b) the HSV color space. Images by Michael Horvath, available under Creative Commons Attribution-Share Alike 3.0 Unported license.

Where M is the transformation matrix and $\vec{\beta}$ is a modifier vector which is designed to capture the information of m. With a more general definition, we expect the general model to do better. Note that, when setting $M = \sigma I_{3\times3}(1 - \alpha_m)$ and $\vec{\beta} = \alpha_m\vec{m}$, this model is exactly the same as our DC model. Moreover, WM18 is also a specific situation of general RGB model where M is a 3×3 identity matrix and $\vec{\beta}$ is the same as $\vec{m}$.

2.2 Modeling in HSV space

Compared with RGB space, when modeling modifiers in HSV space, there are two main differences: angular dimension and modifiers behaviors. As shown in figure 2, HSV space can be represented as a cylindrical geometry with hue as the angular dimension. The hue value starts at the red primary at 0, passes a circle and goes back to red at 2π. As a result, the normal distribution model is not suitable for hue values. For example, when modeling the color "red", the point with hue value 2π equals the point with hue value 0 which can not be learned by the mean square error loss function since the losses of them are totally different.

The other difference is modifier behavior in HSV space. We note that, instead of converging to a point in RGB space, for a certain modifier, vectors from reference colors to target colors are more likely to be parallel with each other.

Since modifiers in HSV space show parallel feature, a modifier m will be modelled as a vector from $\vec{r}$ to $\vec{t}$ in HSV space:

$$\vec{t} = \vec{r} + \vec{m} \tag{5}$$

Here $\vec{m}$ is a function of both m and $\vec{r}$. In addition, comparative modeling will be split into two parts: modeling "hue" dimension as von Mises distribution and other dimensions together as a bivariate

normal distribution. The probability density of the von Mises distribution of the hue dimension is as follows:

$$f(x) = \frac{\exp\left(k\cos\left(x-\mu\right)\right)}{2\pi I_0(k)}. \tag{6}$$

The mean value μ represents the center of hue dimension, k is a measure of distribution shape and $I_0(k)$ is the modified Bessel function of order 0.

When training the model, the parameter k is regarded as a fixed but unknown constant number, thus the loss function that used for training this model, as a result, is as follows:

$$\mathcal{L} = 1 - \frac{1}{n}\sum_{i=1}^{n}\cos\left(h_i - \hat{h}_i\right) \tag{7}$$

where h_i is the hue value of the target color in each instance and $\hat{h}_i$ is the prediction. Because the range of the cosine function is $[-1,\ 1]$, the range of this loss function is $[0,\ 2]$ and the loss equals 0 if and only if $(h_i - \hat{h}_i) \mod 2\pi = 0$ for all i.

2.3 Ensemble model

Based on the different modifier behaviors, using the above models, we define 2 ensemble models. The main idea is some modifiers are easier to represent in RGB space while others are modeled better in HSV space.

Hard Ensemble When treating the problem as binary classification, we apply the above general RGB model and HSV model (see Equation 4 and Equation 5) to get the predictions and convert HSV predictions into RGB space. Then the hard ensemble is trained to predict which color space should be used based on the modifier m and the reference vector $\vec{r}$. The probability of each model that be chosen is:

$$\vec{p} = \text{softmax}(\vec{m}) \tag{8}$$

where $\vec{p}$ is a 2-D vector which indicates the probability of each model being selected and m is the modifier.

Soft Ensemble Alternatively, we can compute a weighted sum over the 2 models. For this model, the final prediction $\hat{t}_i$ is as follows:

$$p_j = \text{softmax}(\vec{m}) \tag{9}$$

$$\hat{t} = \sum_j p_j \hat{t}_j \tag{10}$$

where $\hat{t}_j$ is the prediction of model j for $j \in \{1, 2\}$.

3 Experiments

3.1 Dataset

The dataset used to train the evaluate our model includes 415 triples(reference color label(r), comparative adjective(m), and target color label(t)) in RGB space presented by Winn and Muresan (2018). Munroe (2010) collected original dataset XKCD consists of color description pairs collected in an open online survey and dataset was filtered by McMahan and McMahan and Stone (2015). Winn and Muresan process color labels and convert pairs to triples with 79 unique reference color labels and 81 unique comparatives.

Note that we train models in both RGB and HSV color space but samples in WM18 are only presented in RGB space. In addition, comparatives show the general information between r and t so we use the same approach as Winn and Muresan did: using the mean value of a set of points for a certain color to represent it.

3.2 Experiment Setup

Model configuration: All models in this paper are initialized with Google's pretrained 300-D word2vec embeddings (Mikolov et al., 2013b,a). Those are not updated during training. The single models trained over 800 epochs with batch size 16 and 0.1 learning rate. The hyper-parameters for the two ensemble models are as follows: 2000 epochs, 32 batch size, and 0.1 learning rate.

Architecture: Our models are all multilayer perceptrons. An input modifier is represented as a vector by word2vec pretrained embeddings and followed by two fully connected layers(FC_1 and FC_2) with size 32 and 16 respectively. Let h_1 be the hidden state of FC_2 then $h_1 = FC_2(FC_1(\vec{r},\ \text{word2vec}(m)),\ \vec{r})$. $\vec{r}$ is used as an input for both FC_1 and FC_2. After FC_2, all the other layers are based on hidden state h_1.

Besides the discussed loss functions(Equation 3 and Equation 7), we add one more metric: cosine distances between $(\vec{t} - \vec{r})$ and $\hat{t} - \vec{r}$ which measure the differences in direction.

Evaluation: Following Winn and Muresan (2018), we evaluate the performance in 5 distinct input conditions: (1) Seen Pairings. The triple (r, m, t) has been seen when training models.

	Cosine Similarity ± SD				
Test Condition	General	WM18	HSV	Hard	Reported
Seen Pairings	0.995±0.004	0.977±0.004	0.897±0.062	**0.995±0.005**	0.68
Unseen Pairings	0.810±0.006	0.779±0.005	0.711±0.072	**0.819±0.004**	0.68
Unseen Ref. Color	0.958±0.018	0.801±0.006	0.791±0.031	**0.966±0.002**	0.40
Unseen Comparative	**0.453±0.013**	0.408±0.018	0.341±0.027	0.450±0.001	0.41
Fully Unseen	-0.464±0.199	-0.147±0.077	**0.382±0.514**	-0.606±0.423	-0.21
Overall	0.850±0.002	0.819±0.819	0.750±0.005	**0.852±0.001**	0.65
	Delta-E ± SD				
Overall	6.51±0.079	6.47±0.193	8.73±1.969	**6.37±0.317**	6.8

Table 1: Average cosine similarity score and average Delta-E distance. A smaller Delta-E distance means a less significant difference between two colors. **Bold**: best performance. General: general RGB model. HSV: the model trained in HSV space. Hard: the hard ensemble model. Reported: the original performance which is shown in WM18 paper.

(2) Unseen Pairings. Both r and m have been seen in training data, but not the triple (r, m, t). (3) Unseen Ref. Color. r has not been seen in training, while m has been seen. (4) Unseen Comparative. m has not been seen in training, while r has been seen. (5) Fully Unseen. Neither r nor m have been seen in training.

Because of the small size of the dataset, we report the average performance over 5 runs with different random seeds. Two scores, cosine similarity and Delta-E are applied for evaluating the performance. Cosine similarity measures the difference in terms of vector direction in color space and Delta-E is a non-uniformity metric for measuring color differences.

A small dataset, in addition, leads to another problem: the splitting strategy for training and testing dataset significantly influence the model performance. To ensure a fair comparison to WM18 we evaluate its performance on our own dataset.

3.3 Results and Discussion

As shown in Table 1, our hard ensemble model present clear improvements in terms of both Cosine similarity score and Delta-E distance.

Besides the above models, our DC model is a very specific RGB model where the prediction of $\vec{m}$ does not rely on the reference vector $\vec{r}$. As a result, it gets **0.73** overall cosine similarity score and **6.9** Delta-E. Although the DC model is not as good as WM18, it is useful for explaining the relationship between predictions and convergence behaviour in RGB space. According to Equation 1, α_m does not depend on $\vec{r}$, so by setting $\vec{r}$ as any 2 different vector, we could compute the convergence point and distance scalar.

It is reasonable that the General model achieves the best performance because of the general definition and a larger number of parameters. Compared with WM18, the main improvement of the general model comes from unseen reference color condition which indicates that our model learns the more general pattern of modifiers in RGB space.

On the other hand, modifiers show specific features in different color spaces. Both the soft model and hard ensemble model are designed based on the same basic models but the different ideas leads to significant performance differences: the hard ensemble (**0.852** cosine similarity) is much better than soft ensemble one(**0.813** cosine similarity). The reason why hard ensemble works better is color behavior. Behaviors are more likely to be learned in a certain color space, in this case, the classifier model selects one model based on m and $\vec{r}$ in an instance to make predictions works better than soft ensemble which computes weighted sum of predictions.

4 Conclusion and future work

In this paper, we introduce behaviors of color comparative adjectives, present RGB and HSV models and show a hard ensemble model which achieves state-of-the-art performance. Our model can also learn the convergence points in color space and can be used for making predictions.

One extension is the use of convergence point. We plan to train a model based on XKCD dataset which learns the convergence point for each modifier, and use it as a pre-trained model that can be used as word2vec, given modifiers, our this model

can provide convergence point as inputs for other models.

References

Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Brian McMahan and Matthew Stone. 2015. A bayesian model of grounded color semantics. *Transactions of the Association for Computational Linguistics*, 3:103–115.

Tomas Mikolov, Kai Chen, Greg S. Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

Will Monroe, Robert XD Hawkins, Noah D Goodman, and Christopher Potts. 2017. Colors in context: A pragmatic neural model for grounded language understanding. *Transactions of the Association for Computational Linguistics*, 5:325–338.

Randall Munroe. 2010. Color survey results.

K. van de Sande, T. Gevers, and C. Snoek. 2010. Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1582–1596.

Olivia Winn and Smaranda Muresan. 2018. 'lighter' can still be dark: Modeling comparative color descriptions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 790–795, Melbourne, Australia. Association for Computational Linguistics.